WHAT IS CLAIMED IS:

Ins. A2

1. A context switch controller in a processor that includes an executive storage for holding operand data operated upon by instructions executing on the processor, the executive storage being divided into a plurality of storage groups containing one or more storage elements, the context switch controller comprising:

a dirty bit storage including a plurality of storage bits corresponding to a plurality of respective storage groups in the executive storage;

a dirty bit logic coupled to the dirty bit storage and coupled to receive a destination address field of the instructions, the dirty bit logic responsive to an executed instruction by classifying a destination access as a targeted storage group according to information in the destination address field of the executed instruction and by evaluating the classified destination based on whether the instruction updates the targeted storage group, the dirty bit logic responsive to a context switch by saving storage groups based on the evaluation of the classified destinations.

2. A context switch controller according to Claim 1 wherein:

the executive storage is a register file; and

the dirty bit storage, is a dirty bit register.

3. A context switch controller according to Claim 1 further comprising:

a dirty bit enable storage coupled to the dirty bit storage, coupled to the dirty bit logic, and coupled to the executive storage, the dirty bit enable storage storing a plurality of dirty bits designating enablement or disablement of access to the storage groups in the executive storage on a storage group-by-storage group basis, the dirty bit logic responsive to an instruction that accesses the executive storage by determining whether access is enabled by the dirty bit of the dirty bit enable storage corresponding to the storage group of the executive storage accessed by the instruction, permitting access if access is enabled.

4. A context switch controller according to Claim 3 wherein:
the dirty bit logic is responsive to an instruction that reads the executive storage by determining whether access is enabled by the dirty bit of the dirty bit enable storage corresponding to the storage group of the executive storage read by the instruction, permitting reading if access is enabled.

5. A context switch controller according to Claim 3 wherein:
the dirty bit logic is responsive to an instruction that writes the executive storage by determining whether access is enabled by the dirty bit of the dirty bit enable storage corresponding to the storage group of the executive storage written by the instruction, permitting writing if access is enabled.

6. A context switch controller according to Claim 3 wherein:
the dirty bit logic is responsive to an instruction that accesses the executive storage by determining whether access is enabled by the dirty bit of the dirty bit enable storage corresponding to the storage group of the executive storage accessed by the instruction, generating a trap or exception if access is disabled.

7. A context switch controller according to Claim 3 wherein:
the plurality of dirty bits designating enablement or disablement of access to the storage groups in the executive storage are accessed using a privileged write access instruction.

8. A context switch controller according to Claim 1 wherein:
the dirty bit logic determines whether a storage group in the executive storage has been written since loading of a process or since a context was last restored, the dirty bit logic responsive to a written storage group by

generating a value in the dirty bit storage that designates the written condition of the storage group.

9. A context switch controller according to Claim 1 wherein:
the dirty bit logic saves a storage group when a storage bit designating the storage group in the dirty bit storage indicates that the storage group was write accessed, the dirty bit logic otherwise allowing context switching without saving the storage group.

10. A context switch controller according to Claim 1 wherein:
the dirty bit logic accesses a destination register (rd) field of an instruction and classifies the destination register rd according to the address in the rd field.

11. A processor comprising:
a plurality of functional units;
a register file including a plurality of register file segments coupled to and associated with the plurality of functional units, respectively, the register file being divided into a plurality of register groups;
a dirty bit register coupled to the register file; and
a dirty bit logic coupled to the dirty bit register, the dirty bit logic having a connection for accessing a destination register (rd) field of an instruction and classifying the destination register rd according to the address in the rd field, the classification corresponding to a bit in the dirty bit register, the dirty bit logic evaluating the dirty bit register to designate that the particular classification includes a register that is written by the instruction.

12. A processor according to Claim 11 further comprising:
a dirty bit enable register coupled to the dirty bit register, coupled to the dirty bit logic, and coupled to the register file, the dirty bit enable register storing a plurality of dirty bits designating enablement or disablement

of access to the register file on a register group-by-register group basis, the dirty bit logic responsive to an instruction that accesses the register file by determining whether access is enabled by the dirty bit of the dirty bit enable register corresponding to the register group of the register file accessed by the instruction, permitting access if access is enabled.

13. A processor according to Claim 11 wherein:
the processor is a Very Long Instruction Word (VLIW) processor.

14. A processor according to Claim 11 wherein:
the dirty bit logic classifies the register file into a plurality of groups of registers.

15. A processor according to Claim 11 wherein:
the register file segments are partitioned into local registers and global registers, the global registers being accessible by the plurality of functional units, the local registers being accessible by the functional unit associated with the register file segment containing the local registers.

16. A processor according to Claim 15 wherein:
the dirty bit logic classifies the register file into a plurality of groups of registers including a plurality of register groups within the global register range and one or more register groups in the individual local register groups.

17. A processor according to Claim 15 wherein:
the dirty bit logic classifies the register file into a plurality of groups of registers including a plurality of register groups within the global register range and one register group that includes all of the local register groups.

18. A processor according to Claim 11 wherein:
the dirty bit logic determines whether a register group in the executive register has been written since loading of a process or since a context was last restored, the dirty bit logic responsive to a written register group by generating a value in the dirty bit register that designates the written condition of the register group.

19. A processor according to Claim 11 wherein:
the dirty bit logic saves a register group when a bit designating the register group in the dirty bit register indicates that the register group was write accessed, the dirty bit logic otherwise allowing context switching without saving the register group.

20. A method of switching context in a processor that includes an executive storage for holding operand data operated upon by instructions executing on the processor, the executive storage being divided into a plurality of storage groups containing one or more storage elements, the method comprising:
utilizing a dirty bit storage including a plurality of storage bits corresponding to a plurality of respective storage groups in the executive storage;
receiving a destination address field of the executing instructions;
responsive to an executed instruction, classifying a destination access as a targeted storage group according to information in the destination address field of the executed instruction;
evaluating the classified destination based on whether the instruction updates the targeted storage group; and
responsive to a context switch, saving storage groups based on the evaluation of the classified destinations.

21. A method according to Claim 20 further comprising:
utilizing a dirty bit enable storage;

storing in the dirty bit enable storage a plurality of dirty bits designating enablement or disablement of access to the storage groups in the executive storage on a storage group-by-storage group basis;

responsive to an instruction that accesses the executive storage, determining whether access is enabled by the dirty bit of the dirty bit enable storage corresponding to the storage group of the executive storage accessed by the instruction; and

permitting access if access is enabled.

22. A method according to Claim 21 further comprising:

responsive to an instruction that reads the executive storage, determining whether access is enabled by the dirty bit of the dirty bit enable storage corresponding to the storage group of the executive storage read by the instruction; and

permitting reading if access is enabled.

23. A method according to Claim 21 further comprising:

responsive to an instruction that writes the executive storage,

determining whether access is enabled by the dirty bit of the dirty bit enable storage corresponding to the storage group of the executive storage written by the instruction; and

permitting writing if access is enabled.

24. A method according to Claim 21 further comprising:

responsive to an instruction that accesses the executive storage, determining whether access is enabled by the dirty bit of the dirty bit enable storage corresponding to the storage group of the executive storage accessed by the instruction; and

generating a trap or exception if access is disabled.

25. A method according to Claim 21 further comprising:
accessing the plurality of dirty bits designating enablement or disablement of access to the storage groups in the executive storage using a privileged write access instruction.

26. A method according to Claim 20 further comprising:
determining whether a storage group in the executive storage has been written since loading of a process or since a context was last restored; and
responsive to a written storage group, generating a value in the dirty bit storage that designates the written condition of the storage group.

27. A method according to Claim 20 further comprising:
saving a storage group when a storage bit designating the storage group in the dirty bit storage indicates that the storage group was write accessed; and
otherwise allowing context switching without saving the storage group.

28. A method according to Claim 20 further comprising:
accessing a destination register (rd) field of an instruction; and
classifying the destination register rd according to the address in the rd field.

add C1